

10 Gbps LDPC 编码器的 FPGA 设计

袁瑞佳^{①②} 白宝明^{*①②} 童胜^①

^①(西安电子科技大学综合业务网国家重点实验室 西安 710071)

^②(通信网信息传输与分发技术重点实验室 石家庄 050002)

摘要: 该文针对准循环双对角结构的低密度奇偶校验(LDPC)码,提出了一种基于 FPGA 的高吞吐量编码器实现方法。提出了一种快速流水线双向递归编码算法,能显著提高编码速度;同时设计了一种行间串行列间并行的处理结构计算中间变量,在提高编码并行度的同时可有效减少存储资源的占用量;设计还针对多帧并行编码的情况优化了存储结构,有效复用了数据存储单元和 RAM 地址发生器,进一步提高 FPGA 的资源利用率。对一组码长为 2304 的 IEEE 802.16e 标准 LDPC 码,在 Xilinx XC4VLX40 芯片上,该方法可实现时钟频率 200 MHz,信息吞吐量达 10 Gbps 以上的编码器,且占用不超过 15%的芯片逻辑资源和 50%左右的 RAM 存储资源。

关键词: 低密度奇偶校验(LDPC)码; 编码器; 高吞吐量

中图分类号: TN911.22

文献标识码: A

文章编号: 1009-5896(2011)12-2942-06

DOI: 10.3724/SP.J.1146.2010.01338

FPGA-based Design of LDPC Encoder with Throughput over 10 Gbps

Yuan Rui-jia^{①②} Bai Bao-ming^{①②} Tong Sheng^①

^①(State Key Lab. of Integrated Services Networks, Xidian University, Xi'an 710071, China)

^②(Science and Technology on Information Transmission and Dissemination in Communication Networks Lab, Shijiazhuang 050002, China)

Abstract: This paper presents a high-throughput encoding method for IEEE 802.16e-like Low-Density Parity-Check (LDPC) codes. It is based on a fast double-recursion pipeline method, and can significantly improve the encoding speed. For more parallelism and less storage consumption, a partially-parallel architecture is designed. Furthermore, the storage system is optimized for parallel multi-frame coding, and the data storage unit and RAM address generator are shared for improving resource utilization. Design results are provided for an implementation on a Xilinx XC4VLX40 FPGA for codes with code length 2304 bit. It is shown that the proposed method can achieve a throughput in excess of 10 Gbps under a maximum clock frequency of 200 MHz, with the requirement of no more than 15% gate area and about 50% RAM storage.

Key words: Low-Density Parity-Check (LDPC) code; Encoder; High throughput

1 引言

Gallager^[1]在 1962 年提出的低密度奇偶校验(Low-Density Parity-Check, LDPC)码是一类校验矩阵非常稀疏的线性分组码。文献[2,3]进一步研究表明, LDPC 码采用置信传播算法译码的性能可以逼近 Shannon 容量限。此后, LDPC 码以其优异的性能、较低的译码复杂度和可并行的译码结构,快速成为了信道编码理论的研究热点,并成为了 IEEE802.16e, IEEE 802.11n, IEEE 802.3an 和

DVB-S2 等通信标准建议的信道编码。

如何在大规模集成电路上完成 LDPC 码编译码器的高速实现一直是 LDPC 码应用研究的一个焦点。在编码器的实现上,如果直接采用信息序列与生成矩阵 G 相乘的方法设计编码电路,那么编码所需的数据存储量和计算量都将随着码长的增加以平方速度递增,从而导致编码器在码长较长的情况下因复杂度过高而无法实现。为了实现线性时间复杂度的编码,文献[4]提出了一种简化的编码算法,其思想是将校验矩阵变换成近似下三角的形式来快速计算校验向量,这种算法虽然具有通用性,但其所需的存储和运算量依然过于庞大。文献[5]介绍了两种基于 Richardson-Urbanke 算法的编码器实现。此后,学者们对 LDPC 码快速编码的研究工作转向了

2010-12-06 收到, 2011-09-26 改回

新一代宽带无线移动通信网重大专项(2009ZX03003-011, 010ZX03003-003), 国家自然科学基金(60972046, 61001130)和通信网信息传输与分发技术重点实验室开放课题(ITU-U1007)资助课题

*通信作者: 白宝明 bmbai@mail.xidian.edu.cn

结构化编码方法, 这些方法要求码本身具有特殊的结构特性, 因此会对码的构造产生一定限制, 但這些方法能大幅度降低编码的复杂度和所需的存储空间, 有效解决了编码器在工程上的实现问题。其中文献[6]提出了一种基于图的迭代信息传递编码器, 文献[7]提出了准循环结构的 LDPC 码。对于具有准循环结构的 LDPC 码, 其编码方法可分为两种, 一种是设计生成矩阵具备准循环性质的 LDPC 码, 利用生成矩阵进行编码, 文献[8]给出了这种方法的编码电路; 另一种方法是采用特殊结构的准循环校验矩阵对 LDPC 码进行编码^[9-11], 例如 IEEE 802.16e (WiMAX), IEEE 802.11n(WLAN)等标准采用了一种双对角结构的准循环校验矩阵进行编码, 针对该结构文献[12]提出了一种快速递归编码算法及该算法的快速并行硬件实现架构, 而采用同样的算法, 文献[13]则给出了一种资源占用量极少的串行编码器设计方法。但以上编码器设计都基于长位宽的循环移位电路, 容易引起长路径时延, 且在多帧并行编码时这些方法所需的逻辑资源数量会随着并行处理的数据帧数迅速增加, 因此不适合于大吞吐量的高速编码器设计。

本文对双对角结构的循环移位校验矩阵进行了分析和研究, 在现有的快速递归编码算法基础上提出了一种快速流水线双向递归编码算法, 设计了一种行间串行列间并行的编码器结构并实现了计算校验向量的快速流水线电路, 提出了一种高吞吐量的 LDPC 码编码器实现方案。在 FPGA 上的测试结果表明利用该方法实现 IEEE 802.16e 标准中不同码率的 LDPC 码编码器, 其信息吞吐量均可达到 10 Gbps 以上, 算法具备通用性, 且占用的硬件资源数量少。

2 快速流水线双向递归编码算法

2.1 准循环双对角 LDPC 码

IEEE 802.16e, IEEE 802.11n 等标准中的 LDPC 码是一种准循环双对角 LDPC 码, 它同时具有准循环和双对角两种结构特性。作为一种准循环 LDPC 码, 它的校验矩阵 \mathbf{H} 由多个大小相等的子矩阵构成, 每个子矩阵为全零方阵或单位阵向右循环移位的置换矩阵。若子矩阵的大小为 $z \times z$, 以 -1 表示 $z \times z$ 的全零矩阵, 非负整数 a 表示 $z \times z$ 单位阵循环右移 a 位后的置换阵, 那么一个大小为 $m \times n$ 的准循环校验矩阵 \mathbf{H} 可由一个 $m_b \times n_b$ 的基矩阵 \mathbf{H}_b 唯一表示, \mathbf{H}_b 中的每个元素对应 \mathbf{H} 中的一个子矩阵, 其中码长 $n = z \times n_b$, 校验位个数 $m = z \times m_b$, 信息位个数 $k = z \times k_b$ 。为了实现系统形式的编码, 基矩

阵 \mathbf{H}_b 可分解成两部分 $\mathbf{H}_b = [(\mathbf{H}_{b_1})_{m_b \times k_b} | (\mathbf{H}_{b_2})_{m_b \times m_b}]$, \mathbf{H}_{b_1} 对应系统位部分, \mathbf{H}_{b_2} 对应校验位部分。双对角结构特性是指矩阵 \mathbf{H}_{b_2} 满足如下条件: 将 \mathbf{H}_{b_2} 进一步分解为两部分 \mathbf{h}_b 和 \mathbf{H}'_{b_2} , 如式(1)所示。 \mathbf{h}_b 是 m_b 维的列向量, 其中只含 3 个不等于 -1 的元素, 且有 $h_b(0) = h_b(m_b - 1) \neq -1$, 这里将第 3 个非 -1 元素的行号记为 x 。 \mathbf{H}'_{b_2} 是个双对角阵, 用 i 和 j 表示其行号和列号, 那么当 $i = j$ 和 $i = j + 1$ 时矩阵中对应位置的元素为 0, 其它元素均为 -1 。

$$\mathbf{H}_{b_2} = [\mathbf{h}_b | \mathbf{H}'_{b_2}] = \begin{bmatrix} h_b(0) & 0 & & & \\ -1 & 0 & 0 & -1 & \\ \vdots & & 0 & \ddots & \\ h_b(x) & & \ddots & 0 & \\ \vdots & & -1 & 0 & 0 \\ h_b(m_b - 1) & & & & 0 \end{bmatrix} \quad (1)$$

2.2 快速流水线双向递归编码算法

本文针对准循环双对角 LDPC 码的结构特性, 结合硬件实现上的流水线处理方法, 提出了一种快速流水线双向递归编码算法。该算法能有效提高编码处理的并行度, 且适合在硬件电路上使用流水线方法进行高吞吐量的 LDPC 编码器设计。算法描述如下: 将一个码字 \mathbf{c} 分成输入的信息序列 \mathbf{s} 和编码产生的校验序列 \mathbf{p} 两部分, 将 \mathbf{s} 和 \mathbf{p} 再分解为 k_b 和 m_b 组 z 维子向量, 那么码字可表示为 $\mathbf{c} = [\mathbf{s} | \mathbf{p}] = [\mathbf{s}_0, \mathbf{s}_1, \dots, \mathbf{s}_{k_b-1}, \mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_{m_b-1}]$, 用符号 $\mathbf{L}_{i,j}$ 表示矩阵 \mathbf{H}_{b_1} 中第 i 行第 j 列元素对应的 $z \times z$ 子矩阵, 令 $h_b(0) = h_b(m_b - 1) = l$, $h_b(x) = l_x$, 且令

$$\mathbf{b}_i = \sum_{j=0}^{k_b-1} \mathbf{L}_{i,j} \mathbf{s}_j, \quad 0 \leq i \leq m_b - 1 \quad (2)$$

那么由 $\mathbf{c} \times \mathbf{H}^T = \mathbf{0}$, 可以得到

$$\mathbf{b}_0 + \mathbf{p}_0^{(l)} + \mathbf{p}_1 = \mathbf{0} \quad (3)$$

$$\mathbf{b}_i + \mathbf{p}_i + \mathbf{p}_{i+1} = \mathbf{0}, \quad 1 \leq i \leq m_b - 2, i \neq x \quad (4)$$

$$\mathbf{b}_x + \mathbf{p}_0^{(l_x)} + \mathbf{p}_x + \mathbf{p}_{x+1} = \mathbf{0} \quad (5)$$

$$\mathbf{b}_{m_b-1} + \mathbf{p}_0^{(l)} + \mathbf{p}_{m_b-1} = \mathbf{0} \quad (6)$$

其中 $\mathbf{p}_i^{(l)}$ 表示 \mathbf{p}_i 向右循环移动 l 次后的向量。

将式(3)到式(6)对应于不同行的 m_b 个方程相加可得

$$\mathbf{p}_0^{(l_x)} = \sum_{i=0}^{m_b-1} \mathbf{b}_i \quad (7)$$

求出 \mathbf{p}_0 后, 通过式(3)进行前向递归可求得 \mathbf{p}_1 向量的第 1 个分量 $\mathbf{p}_1(1)$

$$\mathbf{p}_1(1) = \mathbf{b}_0(1) + \mathbf{p}_0^{(l)}(1) \quad (8)$$

符号 $\mathbf{p}_i(j)$ 表示 \mathbf{p}_i 向量的第 j 个分量, 同时可通过式

(6)进行后向递归求得

$$\mathbf{p}_{m_b-1}(1) = \mathbf{b}_{m_b-1}(1) + \mathbf{p}_0^{(1)}(1) \quad (9)$$

求解 $\mathbf{p}_1(1), \mathbf{p}_{m_b-1}(1)$ 后, 继续求解 $\mathbf{p}_1(2), \mathbf{p}_{m_b-1}(2)$, 并通过双向递归求解 $\mathbf{p}_2(1), \mathbf{p}_{m_b-2}(1)$ 有

$$\mathbf{p}_1(2) = \mathbf{b}_0(2) + \mathbf{p}_0^{(1)}(2) \quad (10)$$

$$\mathbf{p}_{m_b-1}(2) = \mathbf{b}_{m_b-1}(2) + \mathbf{p}_0^{(1)}(2) \quad (11)$$

$$\mathbf{p}_2(1) = \mathbf{b}_1(1) + \mathbf{p}_1(1) \quad (12)$$

$$\mathbf{p}_{m_b-2}(1) = \mathbf{b}_{m_b-2}(1) + \mathbf{p}_{m_b-1}(1) \quad (13)$$

以此类推, 直到求解最后出中间的校验向量的最后一个向量

$$\mathbf{p}_{m_b/2}(m_b - k_b) = \mathbf{b}_{m_b/2}(m_b - k_b) + \mathbf{p}_{(m_b/2)+1}(m_b - k_b) \quad (14)$$

采用该快速流水线双向递归编码算法, 求出的校验向量顺序为 $\{\mathbf{p}_0\}, \{\mathbf{p}_1(1), \mathbf{p}_{m_b-1}(1)\}, \{\mathbf{p}_2(2), \mathbf{p}_{m_b-1}(2), \mathbf{p}_2(1), \mathbf{p}_{m_b-2}(1)\}, \{\mathbf{p}_1(3), \mathbf{p}_{m_b-1}(3), \mathbf{p}_2(2), \mathbf{p}_{m_b-2}(2), \mathbf{p}_3(1), \mathbf{p}_{m_b-3}(1)\}, \dots, \{\mathbf{p}_{m_b/2}(m_b - k_b)\}$, 大括号表示多个校验向量的分量可以同时求解, 最后将求出的校验向量与信息向量相结合就可以得到系统形式的码字。采用这种编码算法在硬件实现上具有运算简单, 并行度高, 资源占用少和布线复杂度低的特点。

3 编码器 FPGA 硬件实现技术

以上一节介绍的快速流水线双向递归编码算法为基础, 本文设计了如图 1 所示的高速编码器架构。RAM_MSG, RAM_B, RAM_P0, RAM_P 表示不同的数据存储单元。进行编码操作时, 编码器首先通过式(2)获得编码中间变量 $\mathbf{b}_i(0 \leq i \leq m_b - 1)$ 的值, 计算过程中需要从 RAM_MSG 存储块中读取缓存的信息序列 \mathbf{s} , 期间采用行间串行列间并行的半并行计算方式计算 \mathbf{b}_i , 求得 \mathbf{b}_i 后将结果存入 RAM_B 存储块中供计算 $\mathbf{p}_i(0 \leq i \leq m_b - 1)$ 使用。接着根据式(7)从 RAM_B 中取出各 \mathbf{b}_i 对应的分量值, 顺序计算出 \mathbf{p}_0 向量的各个分量, 结果存入 RAM_P0。最后根据式(8)–式(14), 应用双向递归的快速流水线方法迭代求解出 $\mathbf{p}_i(1 \leq i \leq m_b - 1)$ 后存入 RAM_P 存储块中等待顺序输出。RAM 地址发生器用于产生各存储块的地址和读写信号, 控制数据的有序读写。

3.1 行间串行列间并行的计算结构

基于准循环双对角结构的传统编码器设计方法都是基于长位宽的循环移位电路^[12-14], 采用这种设计方法存在两个主要缺点, 首先长位宽的循环移位电路由于布线复杂, 容易引起长路径时延, 从而导致电路的运行频率较低; 其次, 长位宽循环移位电

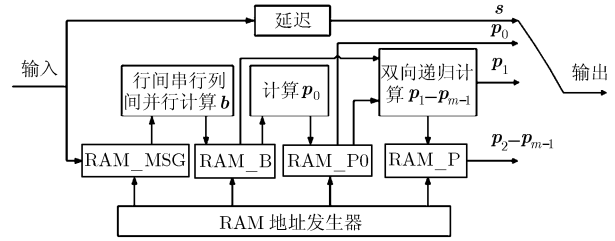


图1 本文设计的编码器架构

路占用的逻辑资源量较大, 在多帧并行处理时, 逻辑资源的占用量会随着并行编码的路数线性递增, 因此它不适合于多帧并行处理的高吞吐量编码器设计。本文在 \mathbf{b}_i 向量的计算上提出了一种行间串行列间并行的计算结构, 信息序列的存储基于 RAM 方式, 采用这种结构设计的编码器电路简单, 占用逻辑资源少, 且多帧信息可存储在同一块 RAM 中, 因此所需的 RAM 资源块数量不会随着并行处理的帧数线性增加, 适合多帧并行处理的高吞吐量编码器设计。在本设计中, 编码器自上而下顺序计算 $z \times m_b$ 行的 \mathbf{b}_i 值, 计算每行对应的 \mathbf{b}_i 分量时, 按照校验矩阵的列分块形式从 k_b 块 RAM 中并行读取取出 k_b 路信息数据进行异或操作, 如图 2 所示, 每路读取的信息数据与其在校验矩阵中的偏移量相对应, 计算下一个 \mathbf{b}_i 分量时只需递增 RAM 的地址或更新至下一行分块的起始偏移地址即可。行间采用串行计算机制主要考虑到多行并行计算时需要同时取出多路信息数据, 这将导致信息位需要重复存储, 从而造成巨大的 RAM 开销。而列分块并行处理则不存在数据冲突问题, 因为不同列块在计算过程中取出的数据对应着信息数据的不同分块。采用上面的结构计算所有的 \mathbf{b}_i 向量共需要 $z \times m_b$ 个时钟周期, 存储信息数据的 RAM 块数量为 k_b 块。这里如果采用行间并行的处理方式, 那么将需要同时从 RAM_MSG 中取出 $k_b \times d_c$ 路数据, d_c 为校验节点的度, 相应地需要 $k_b \times d_c$ 块 RAM 资源。以 IEEE 802.16e 中 5/6 码率的 LDPC 码为例, 在 $d_c=4, k_b=20$ 的情况下, 行列同时并行处理时 RAM_MSG 存储块需要占用 80 块 RAM 资源, 而行间串行列间并行的编码结构仅需要 20 块, 可见与行列双并行处理机制相比, 行间串行列间并行的编计算结构可以节省 d_c 倍的 RAM 资源。

3.2 双向递归快速流水线电路

在求解出所有 \mathbf{b}_i 向量后, 可以根据式(7)计算出校验向量 \mathbf{p}_0 , 接着采用第 2 节介绍的快速流水线双向递归编码算法, 可以依次求出其余的校验向量 $\mathbf{p}_1, \dots, \mathbf{p}_{m_b-1}$ 。根据式(8)–式(14)可利用插入流水线的

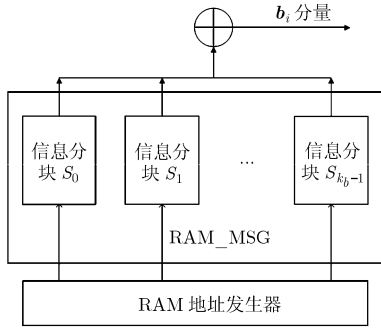


图2 计算 b_i 分量的电路结构图

方法实现近似的同步求解，图3给出了 $m_b = 8$ 时计算校验向量的流水线工作流程，图4给出了校验向量计算的时钟顺序。采用这种快速流水线方法求解 p_1, \dots, p_{m_b-1} 共需要 $z + m_b - 1$ 个时钟周期，仅比 p_i 完全同时求解所需的 z 个时钟周期多出 $m_b - 1$ 个，而在不采用流水线方法的情况下共需要 $z \times m_b$ 个时钟周期才能完成。采用这种流水线结构的另一个好处是可以实现 p_i 的同步输出，即在计算 p_i 校验位完毕后立即输出，不需要对 p_i 向量进行缓存，从而达到节省存储资源的目的。本文实现的编码器就采用了这样的输出结构。

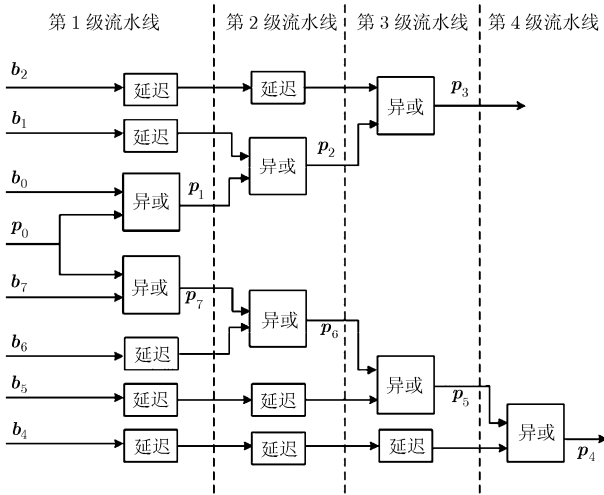


图3 $m_b = 8$ 时计算校验向量的流水线工作流程

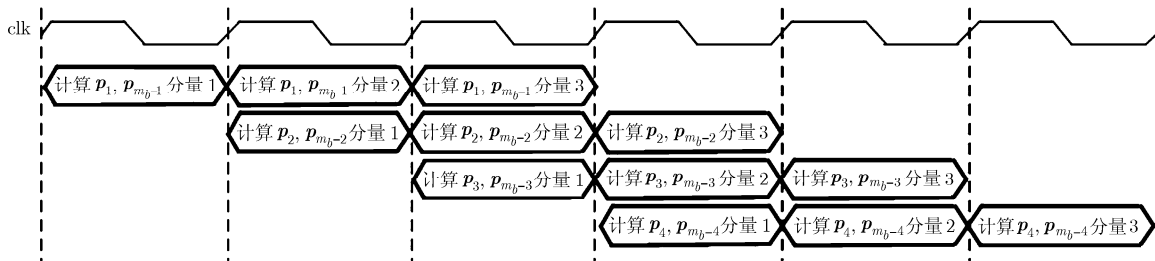


图4 校验向量计算时序图

3.3 存储器管理与设计

FPGA 的硬件资源分为两部分，逻辑资源和 RAM 块(BLOCK RAM, 简称 BRAM)资源。编码器所需的大量数据存储采用 RAM 作为存储媒质更有优势，因为采用逻辑资源存储时，长位宽的信号将引起较大的关键路径延迟，导致编码速度受限，且逻辑资源的占用量与并行处理数据的帧数成正比，不适合多帧并行处理的高速编码器实现。在一片 FPGA 中 RAM 块的数量有限，且存在数据读写上的限制，每块 RAM 在每个时钟周期最多只能读出或写入 2 次数据，因此有效利用好 RAM 块资源，避免使用 RAM 的数量过多，提高 RAM 的利用率十分重要。一种减少 RAM 数量提高利用率的常见方法是将一块 RAM 配置成双读/写的双端口 RAM 作为两块单端口的 RAM 使用，另一种方法是将多路数据合并成一路位宽较长的数据存储在一块 RAM 中，读出后再分路处理。在该编码器实现中这两种技术均被采用。

从上面的讨论可以知道，本设计中应用的存储块分为 4 部分 RAM_MSG, RAM_B, RAM_P0 和 RAM_P，分别用于存储编码器输入信息序列， b_i 的计算结果， p_0 校验向量和 p_2, \dots, p_{m_b-1} 的缓存输出。计算 b_i 时，要求在一个时钟周期内取出 RAM_MSG 中的 k_b 个数据，计算 p_0 时，要求同时取出 RAM_B 内的 m_b 个数据，它们的存储分别使用了 $k_b/2$ 块和 $m_b/2$ 块双端口 RAM。而计算 p_1, p_{m_b-1} 时，只需要从 RAM_P0 中同时取出一个数据，因此 RAM_P0 采用一个单端口 RAM 已足够。 p_2, \dots, p_{m_b-1} 输出缓存，存在 $m_b - 2$ 路数据同时写入的情况，需要 $m_b/2 - 1$ 块双端口 RAM。表1为本文编码器架构所使用的存储器状况。为了提高每块 RAM 资源的利用率，本设计将串行输入的单帧数据通过串/并变换为多帧信息数据，编码器对多帧并行数据进行同步处理，处理时将多帧并行数据对应的比特合并存储在同一存储块中。在硬件实现部分，我们采用了 Xilinx Virtex4 系列的 FPGA，它提供 18 kbit 的 RAM，36 帧并行输入时，在使用 RAM 资源时，

表1 本文编码器架构所使用的存储器数量

存储块名称	类型	占用RAM块数量
RAM_MSG	双端口RAM	$k_b/2$
RAM_B	双端口RAM	$m_b/2$
RAM_P0	单端口RAM	1
RAM_P	双端口RAM	$m_b/2 - 1$
总计	Block RAM	$k_b/2 + m_b$

需要将位宽配置为 36，深度配置为 192，占用 1 块 18 kbit 的 RAM 资源。108 帧并行输入时，位宽为 108，深度为 192，需要 3 块 18 kbit 的 RAM 资源，深度为 96，需要两块 RAM 资源，RAM_P0 所需的 3 块单端口 RAM 可使用 1 块双端口 RAM 加 1 块单端口 RAM 代替。由算法描述部分可知编码时只需要数据间的异或操作，因此可以将合并后的多比特数据直接进行编码计算来等效多路编码器并行计算，经过多比特数据编码后再通过并/串变换模块输出。采用这种合并数据的存储方式的另一个突出优点是可使多路编码器共享同一 RAM 地址发生器，避免不同的数据帧处理时重复产生多路地址，减少逻辑资源的占用量。

4 硬件实现结果

采用本文提出的编码器设计方法对 IEEE 802.16e 标准中 6 种不同码率的 LDPC 码进行实现，芯片使用 Xilinx 公司的 XC4VLX40 FPGA，在 ISE10.1 平台上进行逻辑综合和布局布线，在 ModelSim 6.2 上进行功能仿真和时序仿真，几种编码器实现的硬件资源占用情况和吞吐量如表 2 所示。不同码率的编码器实现在采用 36 bit 并行输入和 108 bit 并行输入两种情况下均可满足 5 ns 的布局布线约束，工作时钟频率可达 200 MHz。由表 2 的数据可见几种编码器在并行处理的数据帧数相同的情况下所占用的硬件资源也相近，在 36 bit 并行输入时，6 种编码器所需的逻辑资源约占芯片逻辑资源的 8%，而 RAM 资源的占用量最多不超过 19%；在 108 bit 并行输入时，编码器所需的逻辑资源和 RAM 资源均不超过 15% 和 55%，且编码器的输入总吞吐量不低于 10 Gbps。可见，该编码器设计方法具有逻辑资源需求量少和信息吞吐量高的特点。

表 3 是本文编码器实现结果和文献[15,16]实现结果的比较。为了比较的公平性，我们从 3 种实现方法中取出 3 个码长和码率相近的实现进行对比。从表 3 的数据可见，本设计所使用的 Slice 资源数量为 2852，少于文献[15]的 12306 和文献[16]的 4525，且编码器的输入吞吐量可以做得更高。在 RAM 资

源的占用上，文献[15]采用 k_b 块 z 比特移位寄存器并行处理的结构来计算校验向量，当码长为 2304，码率为 5/6 时， $k_b=20$ ， $z=96$ ，此时编码器需要 20 路位宽为 96 的存储块作为信息输入缓存。采用与本文相同的 FPGA，在位宽为 96 的情况下，每路信息缓存需要 3 块 RAM 资源，应用与本文相同的 RAM 优化技术可将 RAM 数量降低到一半，因此信息存储部分共需要 $20 \times 3/2=30$ 块 RAM，而校验位的输出缓存部分则需要 $m_b=4$ 路位宽为 96 的存储块，共需 $4 \times 3/2=6$ 块 RAM，因此文献[15]的编码器实现共需要 36 块 RAM 资源。文献[16]采用混合方式来计算校验向量，校验向量的第 1 部分 p_1 采用与 G 矩阵直接相乘的方法得到，另一部分 p_2 则由 RU 算法^[5]计算得到。采用这样的计算方法，编码器必须存储 G 矩阵和 RU 算法所需的 A 矩阵、B 矩阵和 T 矩阵^[6]，其中各矩阵相乘时均需要进行 324 bit 的运算操作，信息位缓存、G 矩阵、A 矩阵、B 矩阵和 T 矩阵的存储各需要 9 块 RAM，共 $5 \times 9=45$ 块。另外，文献[16]中编码器的输出缓存共需要 3 块 RAM，分别用于信息位的延时输出， p_1 缓存输出和 p_2 缓存输出，因此该编码器实现所需的 RAM 资源总共为 48 块。从 RAM 数量的对比上可以看到，本文的编码器实现较文献[16]所需的 RAM 资源更少，但与文献[15]相比则需要更多一些。这主要是由于文献[15]对校验向量的计算不需要任何 RAM 来缓存中间结果，而本文的计算结构则要求对中间变量 b_i 的结果作缓存处理，但这种处理方式也使得在整个编码器架构中不再需要大量的长位宽位移寄存器，从而让编码器所需的逻辑资源更少且运行频率更高，进而带来更低的功耗和更大的系统吞吐量。如果从 RAM 利用率的角度进行对比的话，本文所提出的编码器实现架构则较文献[15,16]的实现方法更具优势。

5 结论

本文针对 IEEE 802.16e 等标准中的结构化低密度奇偶校验码，利用其校验矩阵 H 的准循环双对角结构和分块特性，设计了一种高吞吐量的快速编码实现方法。在该编码器实现中我们提出了一种行间串行列间并行的运算结构，设计了一种校验向量的快速流水线计算方法，同时针对多帧并行处理优化了编码器的 RAM 存储结构。在基于 Xilinx 公司的 XC4VLX40 FPGA 上的实现结果表明该设计方法能使用较少的 FPGA 资源完成 LDPC 码的快速并行编码，达到 10 Gbps 量级的吞吐量。另外，在不同码型和码率上的实现结果表明该设计方法对准循环双对角 LDPC 码编码器的设计具有通用性。

表2 IEEE 802.16e协议中码长为2304, z=96的编码器耗费FPGA资源及吞吐量

码率	36 bit并行输入			108 bit并行输入		
	Slice资源	RAM资源	吞吐量(Gbps)	Slice资源	RAM资源	吞吐量(Gbps)
1/2	1467(7%)	18(19%)	3.6	2619(14%)	53(55%)	10.8
2/3A	1530(8%)	16(17%)	4.8	2682(14%)	47(49%)	14.4
2/3B	1519(8%)	16(17%)	4.8	2671(14%)	47(49%)	14.4
3/4A	1775(9%)	15(16%)	5.4	2927(15%)	44(46%)	16.2
3/4B	1731(9%)	15(16%)	5.4	2883(15%)	44(46%)	16.2
5/6	1700(9%)	14(15%)	6	2852(15%)	41(43%)	18

表3 本文与文献[15,16]实现的编码器比较

LDPC码	码率	Slice资源	RAM资源	时钟频率(MHz)	吞吐量(Gbps)
文献[15](2304, 1920)码	0.83	12306	36	150.69	5.67
文献[16](2048, 1723)码	0.84	4525	48	215.24	10
本文(2304, 1920)码	0.83	2852	41	200.20	18

参考文献

- [1] Gallager R G. Low-density parity-check codes [J]. *IRE Transactions on Information Theory*, 1962, 8(1): 21-28.
 - [2] MacKay D J C and Neal R M. Near Shannon limit performance of low density parity check codes [J]. *Electronics Letters*, 1996, 32(18): 1645-1646.
 - [3] Spielman D A. Linear-time encodable and decodable error-correcting codes [J]. *IEEE Transactions on Information Theory*, 1996, 42(11): 1723-1731.
 - [4] Richardson T J and Urbanke R L. Efficient encoding of low-density parity-check codes [J]. *IEEE Transactions on Information Theory*, 2001, 47(2): 638-656.
 - [5] Lee D U, Luk W, et al. A flexible hardware encoder for low-density parity-check codes [C]. *Field-Programmable Custom Computing Machines(FCCM)*, Napa, 2004: 101-111.
 - [6] Shaqfeh M and Goertz N. Systematic modification of parity-check matrices for efficient encoding of LDPC codes [C]. *IEEE International Conference on Communications (ICC)*, Glasgow, 2007: 945-950.
 - [7] Kou Y, Lin S, Fossorier, et al. Low-density parity-check codes based on finite geometries: a rediscovery and new results [J]. *IEEE Transactions on Information Theory*, 2001, 47(7): 2711-2736.
 - [8] Li Z, Chen L, Zeng L, et al. Efficient encoding of quasi-cyclic low-density parity-check codes [J]. *IEEE Transactions on Communications*, 2006, 54(1): 71-81.
 - [9] Wai M Tam, Francis C M Lau, and Chi K Tse. A Class of QC-LDPC codes with low encoding complexity and good error performance[J]. *IEEE Communications Letters*, 2010, 14(2): 169-171.
 - [10] Pérez J M and Fernández V. 3GPP2/802.20 RC/QC-LDPC Encoding[C]. *European Wireless Conference*, Lucca, 2010: 157-162.
 - [11] Jung Y and Kim J. Memory-efficient and high-speed LDPC encoder[J]. *Electronics Letters*, 2010, 46(14): 1035-1036.
 - [12] Perez J M and Fernandez V. Low-cost encoding of IEEE 802.11n [J]. *Electronics Letters*, 2008, 44(4): 1411-1412.
 - [13] Cai Z, Hao J, Tan P H, et al. Efficient encoding of IEEE 802.11n LDPC codes [J]. *Electronics Letters*, 2006, 42(25): 1471-1472.
 - [14] Ma Zhuo, Li Ying, and Wang Xin-mei. A quasi-parallel encoder of quasi-cyclic LDPC codes in IEEE 802.16e[C]. *Information Science and Engineering (ICISE)*, Nanjing, 2009: 2492-2495.
 - [15] Sunitha K and Don M. Implementation of a flexible encoder for structured low-density parity-check codes [C]. *Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, Victoria, 2007: 438-441.
 - [16] Aaron E C and Keshab K P. A low-complexity hybrid LDPC code encoder for IEEE 802.3an (10GBase-T) Ethernet [J]. *IEEE Transactions on Signal Processing*, 2009, 57(10): 4085-4094.
- 袁瑞佳: 男, 1982年生, 博士生, 研究方向为信道编码与调制。
 白宝明: 男, 1966年生, 教授, 博士生导师, 研究方向为信息与编码理论、编码调制技术和无线通信、量子通信与编码。
 童胜: 男, 1979年生, 博士, 讲师, 研究方向为信息与编码理论、信道编码与调制。